COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Spatial Domain Processing and Image Enhancement

Lecture 4, Feb 18th, 2008

Lexing Xie

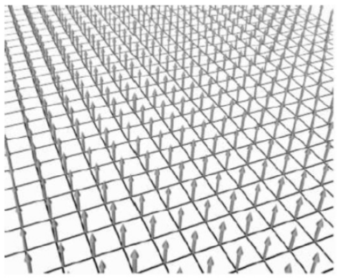
EE4830 Digital Image Processing
http://www.ee.columbia.edu/~xlx/ee4830/

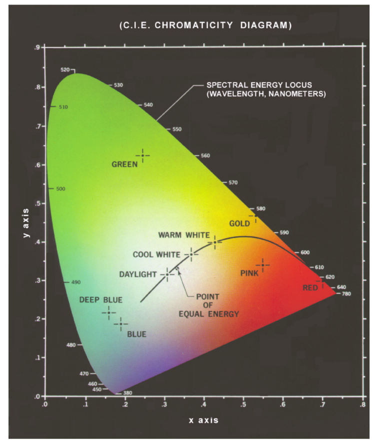
thanks to Shahram Ebadollahi and Min Wu for slides and materials

announcements

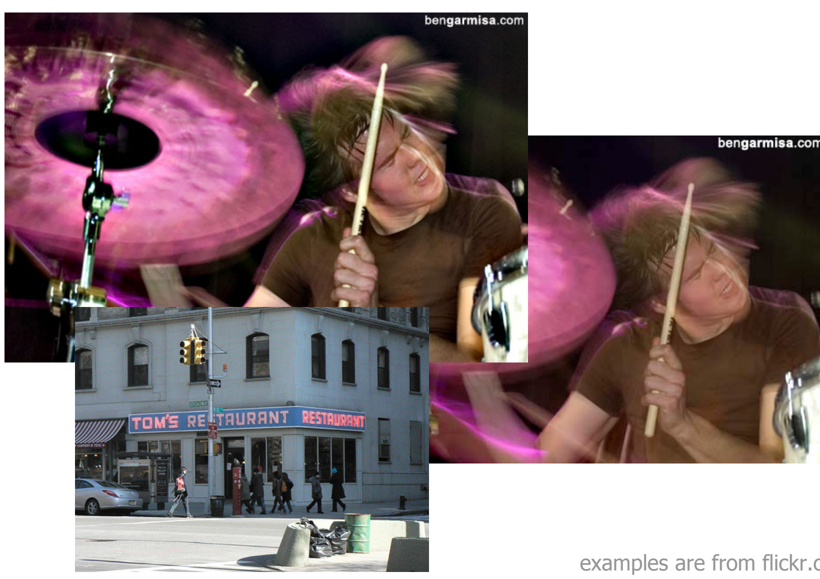
- Today
  - HW1 due
  - HW2 out

recap





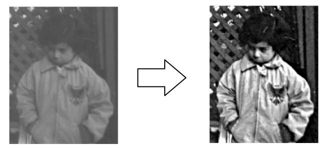
why spatial processing



examples are from flickr.com

roadmap for today

- Application



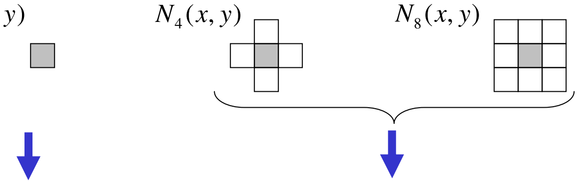
- Method

$$f \xrightarrow{T_N(.)} g = T_N(f)$$

$f(x,y) \quad , \quad 1 \le x \le M, 1 \le y \le N$ $\qquad$ $g(x,y) \quad , \quad 1 \le x \le M, 1 \le y \le N$

$T_N(.)$: Spatial operator defined on a neighborhood *N* of a given pixel

$N_0(x,y)$ $\qquad$ $N_4(x,y)$ $\qquad$ $N_8(x,y)$



point processing $\qquad$ ***mask/kernel processing***

outline

- What and why
  - Spatial domain processing for image enhancement
- Intensity Transformation
- Spatial Filtering

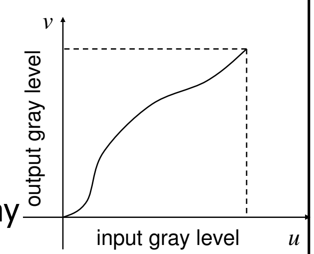
intensity transformation / point operation

- Map a given gray or color level *u* to a new level v

$$f(x,y) \rightarrow g(x,y) \qquad v = \mathcal{T}(u)$$

$$x = 1,\ldots,M,\ y = 1,\ldots,N \qquad u,v = 0,\ldots,255$$

- Memory-less, direction-less operation
  - output at (x, y) only depend on the input intensity at the same point
  - Pixels of the same intensity gets the same transformation
- Does not bring in new information, may cause loss of information
- But can improve visual appearance or make features easier to detect



intensity transformation / point operation

- Two examples we already saw

  Color space transformation

  Scalar quantization

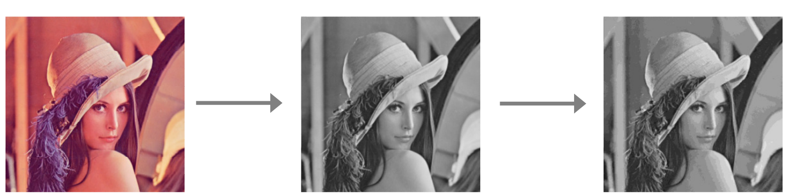
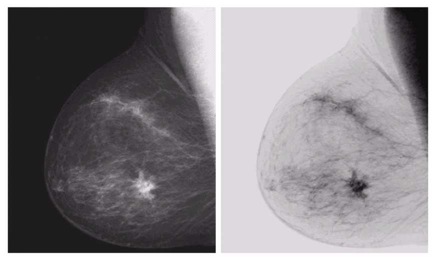


image negatives

$v = 255 - u \qquad u, v = 0, \ldots, 255$

the appearance of photographic negatives

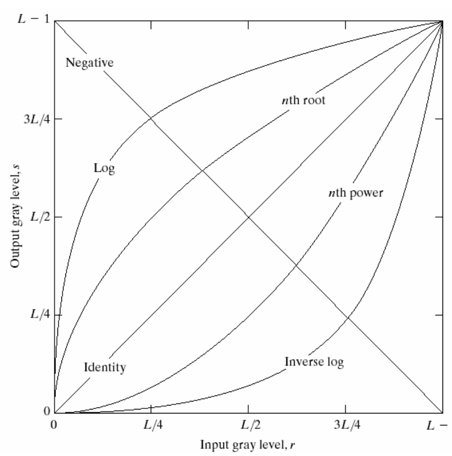
- Enhance white or gray detail on dark regions, esp. when black areas are dominant in size



a b
FIGURE 3.4 (a) Original digital mammogram. (b) Negative image obtained using the negative transformation in Eq. (3.2-1). (Courtesy of G.E. Medical Systems.)

basic intensity transform functions

FIGURE 3.3 Some basic gray-level transformation functions used for image enhancement.



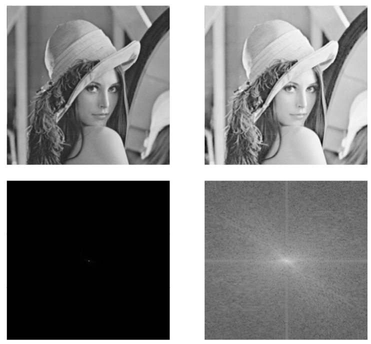
- monotonic, reversible
- *compress* or *stretch* certain range of gray-levels

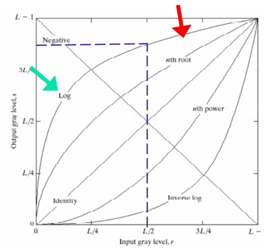
log transform

$$v = c\log(1+u)$$

lena

FFT(lena)





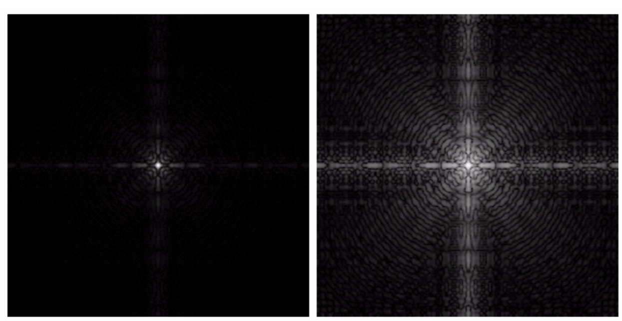
stretch:
u ∈ [0, .5] →
v ∈ [0, .59]

compress:
u ∈ [.5, 1] →
v ∈ [.59, 1]

```
im = imread('lena.png')
a = abs(fftshift(fft2(double(im))));
c = log(1+double(im)); c = range_normalize(c);
b = log(1+a); b=b/max(b(:));
```


a b

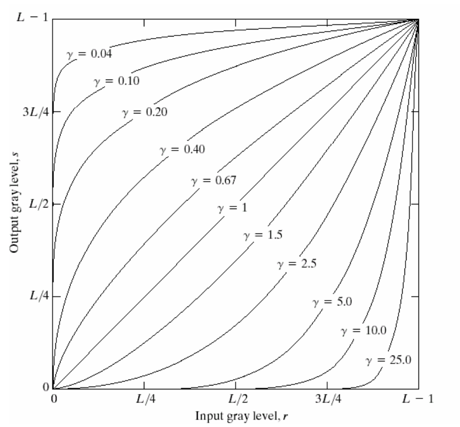
FIGURE 3.5
(a) Fourier spectrum.
(b) Result of applying the log transformation given in Eq. (3.2-2) with $c = 1$.



power-law transformation

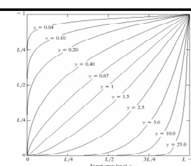
$$v = c \cdot u^{\gamma}$$

- power-law response functions in practice
  - CRT Intensity-to-voltage function has $\gamma \approx 1.8 \sim 2.5$
  - Camera capturing distortion with γ_c = 1.0-1.7
  - Similar device curves in scanners, printers, …

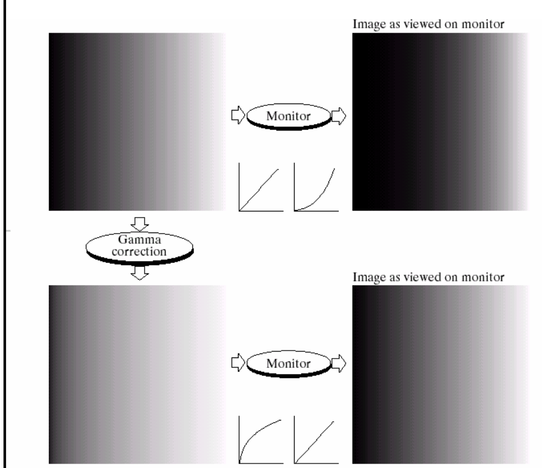


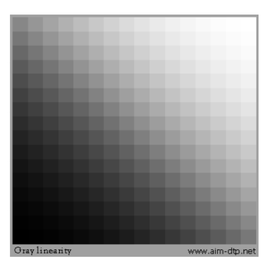
- power-law transformations are also useful for general purpose contrast manipulation

gamma correction



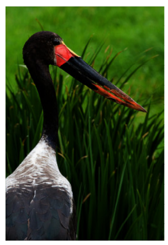
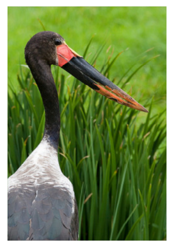
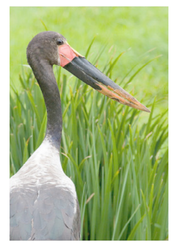
- make linear input appear linear on displays
- method: calibration pattern + interactive adjustment



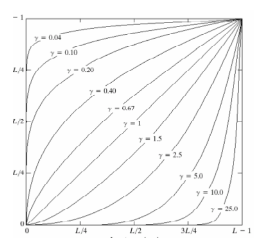


example calibration chart

effect of gamma on consumer photos

$L_0^{2.2}$	L_0	$L_0^{1/2.2}$
		
		

what gamma to use?



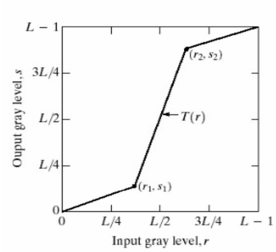
$\gamma > 1$

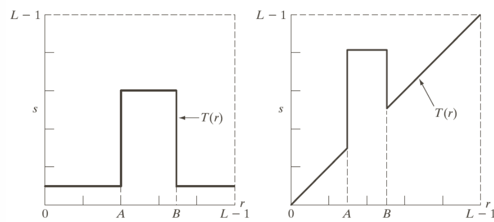
$\gamma < 1$

?

more intensity transform

- log, gamma … closed-form functions on [0,1]
- can be more flexible



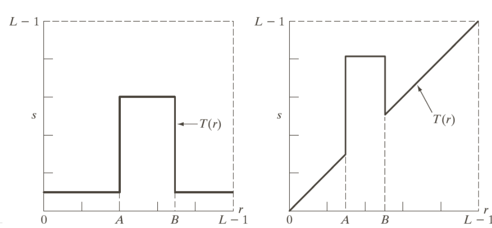


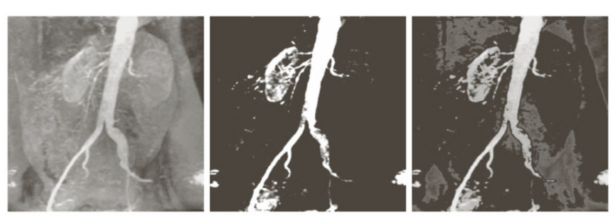
contrast stretching

intensity slicing

a b

FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.





a b c

FIGURE 3.12 (a) Aortic angiogram. (b) Result of using a slicing transformation of the type illustrated in Fig. 3.11(a), with the range of intensities of interest selected in the upper end of the gray scale. (c) Result of using the transformation in Fig. 3.11(b), with the selected area set to black, so that grays in the area of the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)

image bit-planes

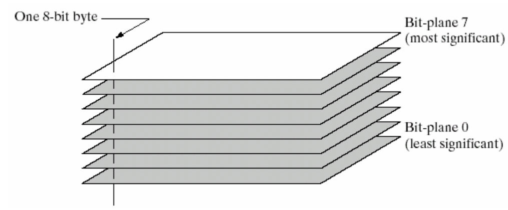
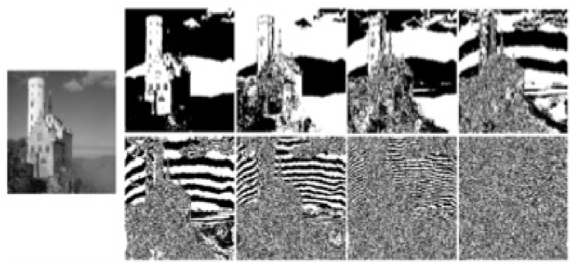


FIGURE 3.12 Bit-plane representation of an 8-bit image.

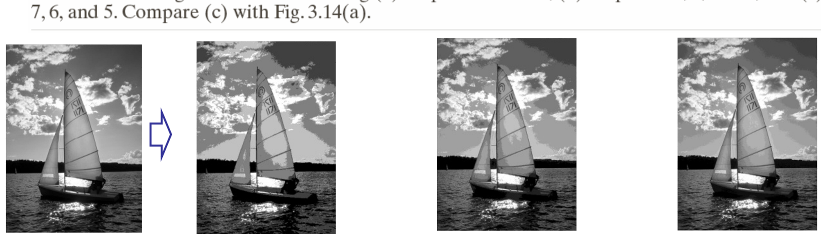


slicing bitplanes

- Depend on relative importance of bits
- How much to slice depend on image content
- Useful in image compression, e.g. JPEG2000

a b c

FIGURE 3.15 Images reconstructed using (a) bit planes 8 and 7; (b) bit planes 8, 7, and 6; and (c) bit planes 8, 7, 6, and 5. Compare (c) with Fig. 3.14(a).



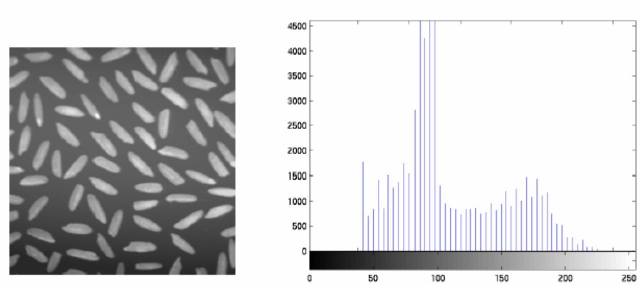
outline

- What and why
  - Image enhancement
  - Spatial domain processing
- Intensity Transformation
  - Intensity transformation functions (negative, log, gamma), intensity and bit-place slicing, contrast stretching
  - Histograms: equalization, matching, local processing
- Spatial Filtering
  - Filtering basics, smoothing filters, sharpening filters, unsharp masking, laplacian
- Combining spatial operations

gray-level image histogram

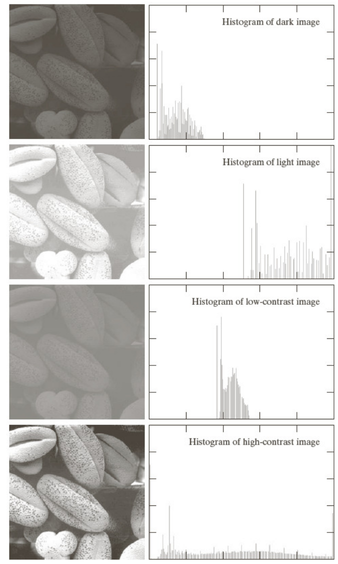
- Represents the relative frequency of occurrence of the various gray levels in the image
  - For each gray level, count the number of pixels having that level
  - Can group nearby levels to form a big bin & count #pixels in it

```
I = imread('rice.tif');
imshow(I)
figure, imhist(I,64)
```



interpretations of histogram

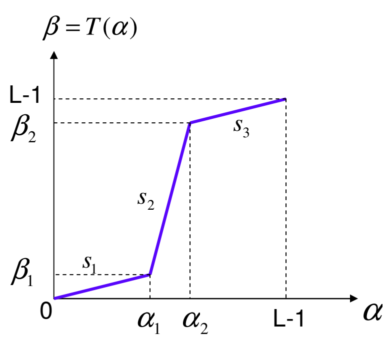
- if pixel values are i.i.d random variables → histogram is an estimate of the probability distribution of the r.v.
- "unbalanced" histograms do not fully utilize the dynamic range
  - Low contrast image: narrow luminance range
  - Under-exposed image: concentrating on the dark side
  - Over-exposed image: concentrating on the bright side
- "balanced" histogram gives more pleasant look and reveals rich details

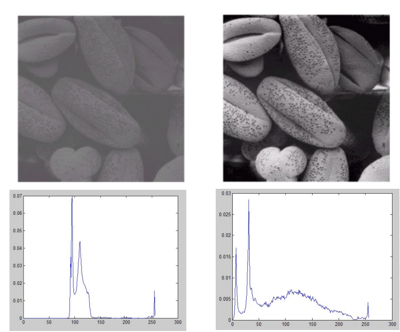


contrast stretching

Stretch the over-concentrated gray-levels

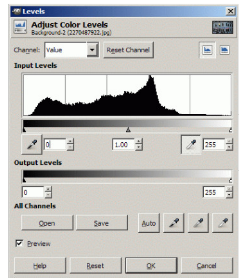
Piece-wise linear function, where the slope in the stretching region is greater than 1.





-25-

… in practice



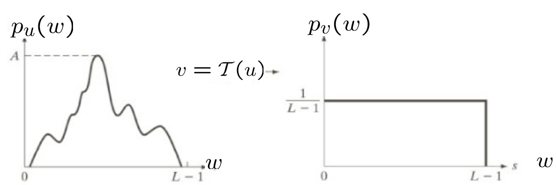
- intuition about a “good” image:
  - a “uniform” histogram spanning a large variety of gray tones
- can we figure out a stretching function automatically?

-26-

histogram equalization

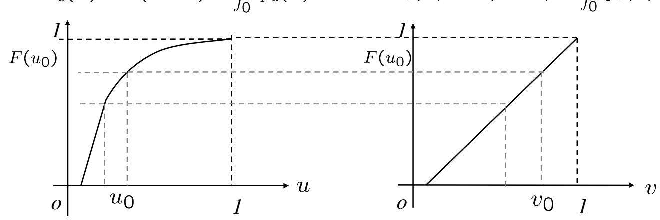
- goal: map the each luminance level to a new value such that the output image has approximately uniform distribution of gray levels
- two desired properties
  - monotonic (non-decreasing) function: no value reversals
  - [0,1]→[0.1] : the output range being the same as the input range

pdf



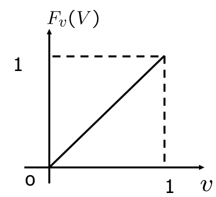
$F_u(U) = P(U < u) = \int_0^u p_u(w)dw$ $\quad$ $F_v(V) = P(V < v) = \int_0^v p_v(w)dw$

cdf



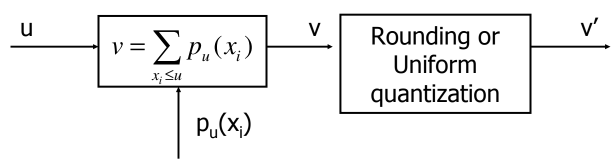
histogram equalization

- make
  $$v = F_u(u) = P(U < u) = \int_0^u p_u(w)dw$$
- show
  $$F_v(V) = v, \ \ v \in [0, 1]$$



$$F_v(V) \ = P(V < v) \ = P(F_u(U) < v)$$
$$= P(U < F_u^{-1}(v)) \ = F_u(F_u^{-1}(v)) \ = v$$

implementing histogram equalization



compute histogram $$p_u(x_i) = \frac{n(x_i)}{\sum_{i=0}^{L-1} n(x_i)} \quad \text{for i} = 0, ..., \text{L - 1}$$

equalize $$v = (L-1)\sum_{x_i=0}^{u} p_u(x_i)$$

or $$v = \frac{L-1}{MN}\sum_{x_i=0}^{u} n(x_i)$$

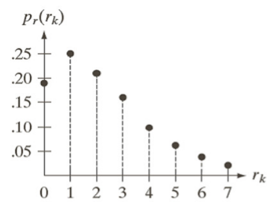
round the output $$v' = round(v)$$

- Only depend on the input image histogram
- Fast to implement
- For *u* in discrete prob. distribution, the output *v* will be approximately uniform

a toy example

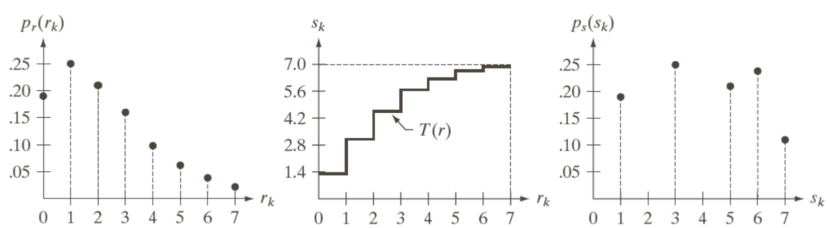
$$v = (L-1)\sum_{x_i=0}^{u} p_u(x_i)$$

r_k	n_k	$p_r(r_k) = n_k/MN$	$7 * \sum_u p(u)$	v
$r_0 = 0$	790	0.19		
$r_1 = 1$	1023	0.25		
$r_2 = 2$	850	0.21		
$r_3 = 3$	656	0.16		
$r_4 = 4$	329	0.08		
$r_5 = 5$	245	0.06		
$r_6 = 6$	122	0.03		
$r_7 = 7$	81	0.02		



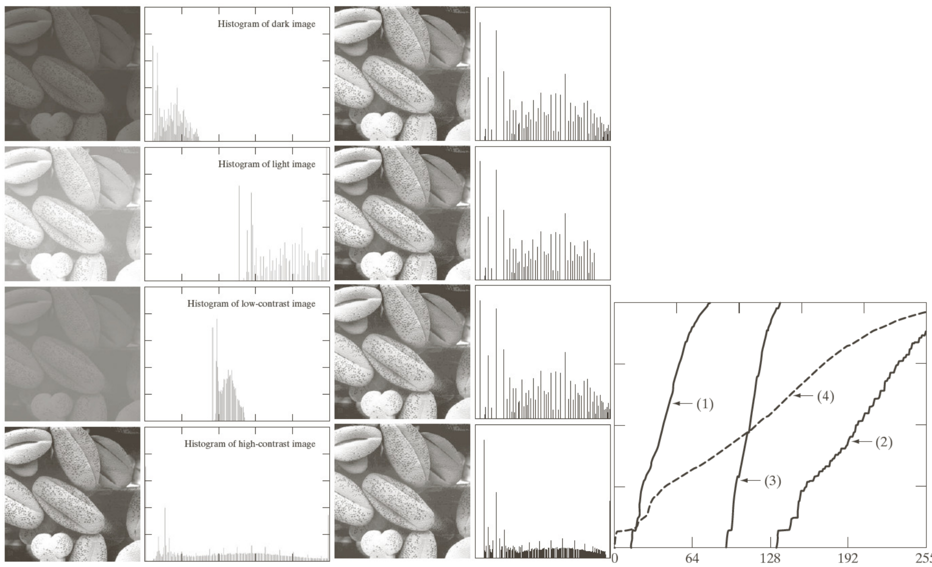
a toy example

r_k	n_k	$p_r(r_k) = n_k/MN$	$7 * \sum_u p(u)$	v
$r_0 = 0$	790	0.19	1.33	1
$r_1 = 1$	1023	0.25	3.08	3
$r_2 = 2$	850	0.21	4.55	5
$r_3 = 3$	656	0.16	5.67	6
$r_4 = 4$	329	0.08	6.23	6
$r_5 = 5$	245	0.06	6.65	7
$r_6 = 6$	122	0.03	6.86	7
$r_7 = 7$	81	0.02	7.00	7

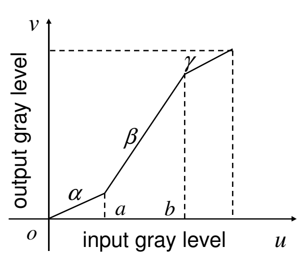


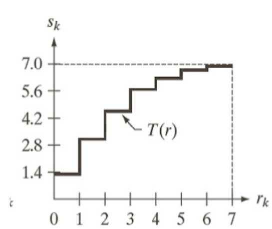
a b c

histogram equalization example



contrast-stretching vs. histogram equalization





- function form
- reversible? loss of information?
- input/output?
- automatic/interactive?

histogram matching

- Histogram matching/specification
  - Want output v with specified p.d.f. $p_V(v)$
  - Use a uniformly distributed random vairable W as an intermediate step
    - $W = F_U(u) = F_V(v) \rightarrow V = F^{-1}_V (F_U(u))$
  - Approximation in the intermediate step needed for discrete r.v.
    - $W_1 = F_U(u)$, $W_2 = F_V(v) \rightarrow$ take v s.t. its w2 is equal to or just above w1

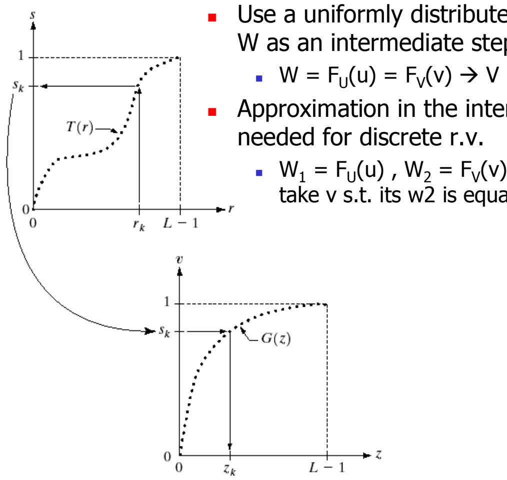
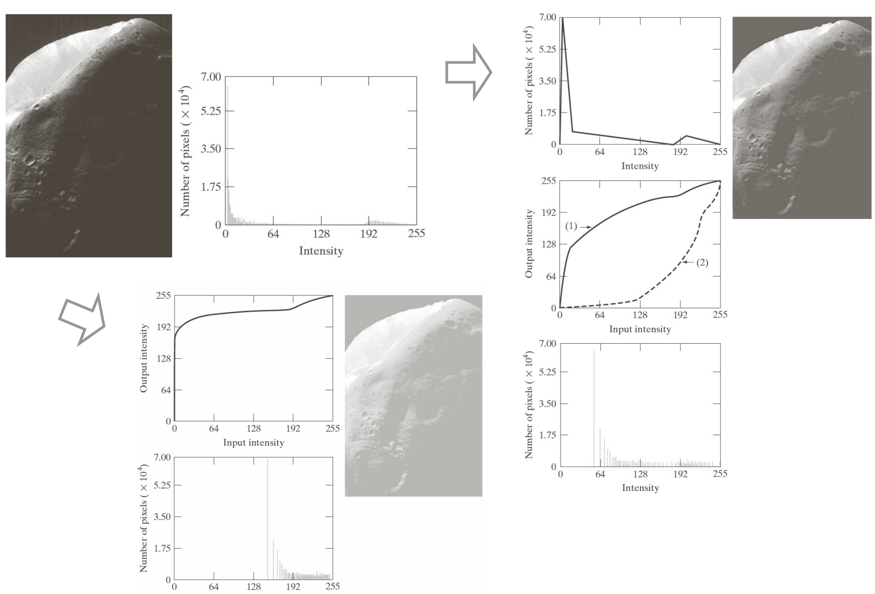


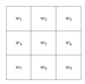
FIGURE 3.19 (a) Graphical interpretation of mapping from r_k to s_k via $T(r)$. (b) Mapping of z_q to its corresponding value v_q via $G(z)$. (c) Inverse mapping from s_k to its corresponding value of z_k.

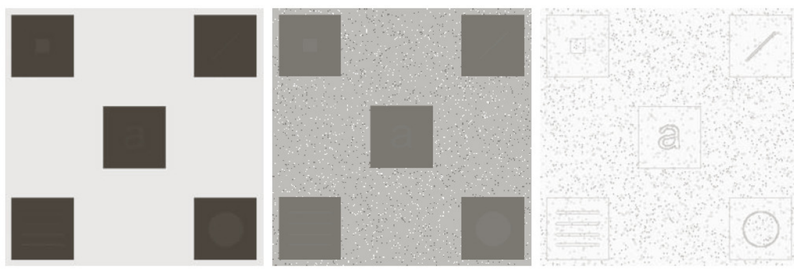
histogram matching example



local histogram processing

- problem: global spatial processing not always desirable
- solution: apply point-operations to a pixel neighborhood with a sliding window





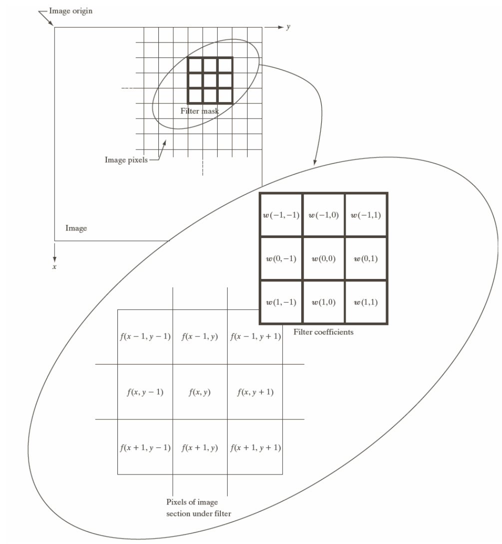
a b c

FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.

outline

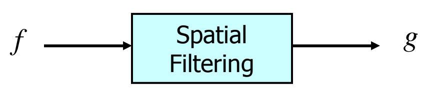
- What and why
  - Image enhancement
  - Spatial domain processing
- Intensity Transformation
  - Intensity transformation functions (negative, log, gamma), intensity and bit-place slicing, contrast stretching
  - Histograms: equalization, matching, local processing
- Spatial Filtering
  - Filtering basics, smoothing filters, sharpening filters, unsharp masking, laplacian
- Combining spatial operations (sec. 3.7)

spatial filtering in image neighborhoods

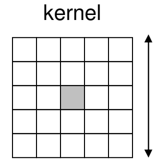


kernel operator / filter masks

$$T_N(.) = w(.)$$



kernel



$$g(m,n) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} w(i,j) f(m+i, n+j)$$

$$1 \le m \le M$$
$$1 \le n \le N$$

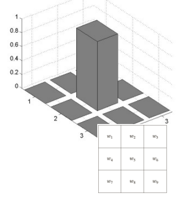
Smoothing: Image Averaging

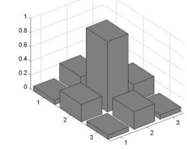
$\frac{1}{9} \times$

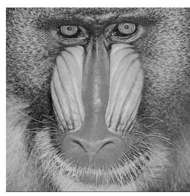
1	1	1
1	1	1
1	1	1

$\frac{1}{16} \times$

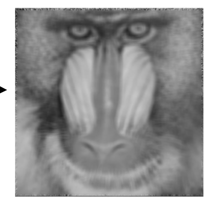
1	2	1
2	4	2
1	2	1







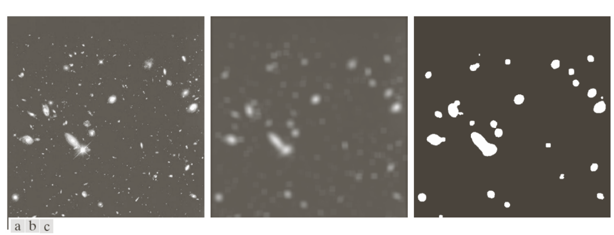
smoothing operator



Low-pass filter, leads to softened edges

spatial averaging can suppress noise

- image with iid noise y*(m,n) = x(m,n) + N(m,n)*
- averaging
  v*(m,n) = (1/N$_w$) Σ x(m-k, n-l) + (1/N$_w$) Σ N(m-k, n-l)*
    - N_w: number of pixels in the averaging window
  - Noise variance reduced by a factor of N_w
  - SNR improved by a factor of N_w
  - Window size is limited to avoid excessive blurring



a b c

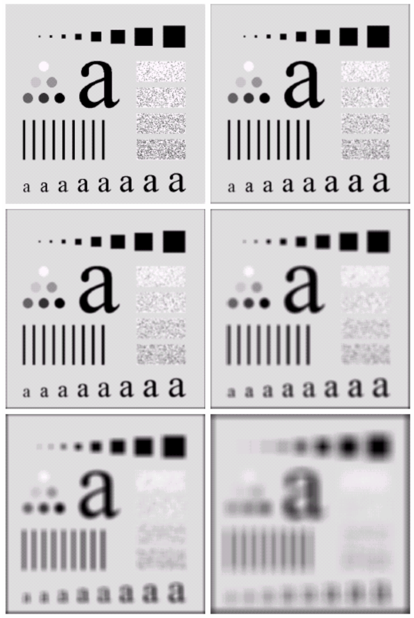
FIGURE 3.34 (a) Image of size 528 × 485 pixels from the Hubble Space Telescope. (b) Image filtered with a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

smoothing operator of different sizes

original | 3x3

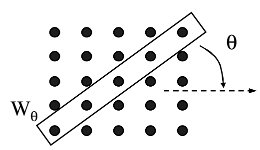
5x5 | 9x9

15x15 | 35x35



directional smoothing

- Problems with simple spatial averaging mask
  - Edges get blurred
- Improvement
  - Restrict smoothing to along edge direction
  - Avoid filtering across edges



- Directional smoothing
  - Compute spatial average along several directions
  - Take the result from the direction giving the smallest changes before & after filtering
- Other solutions
  - Use more explicit edge detection and adapt filtering accordingly

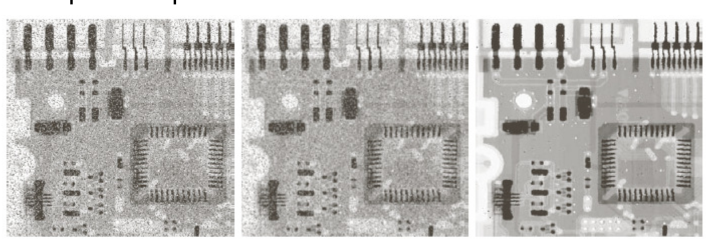
non-linear smoothing operator

- Median filtering
  - median value ξ over a small window of size N_w

    $$\tilde{x} = sort(x);\ \ \xi = \tilde{x}[\frac{N_w + 1}{2}]$$
  - nonlinear
    - median{ x(m) + y(m) } ≠ median{x(m)} + median{y(m)}
  - odd window size is commonly used
    - 3x3, 5x5, 7x7
    - 5-pixel "+"-shaped window
  - for even-sized windows take the average of two middle values as output
- Other order statistics: min, max, x-percentile …

median filter example

- Median filtering
  - resilient to statistical outliers
  - incurs less blurring
  - simple to implement



a b c

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3 × 3 averaging mask. (c) Noise reduction with a 3 × 3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

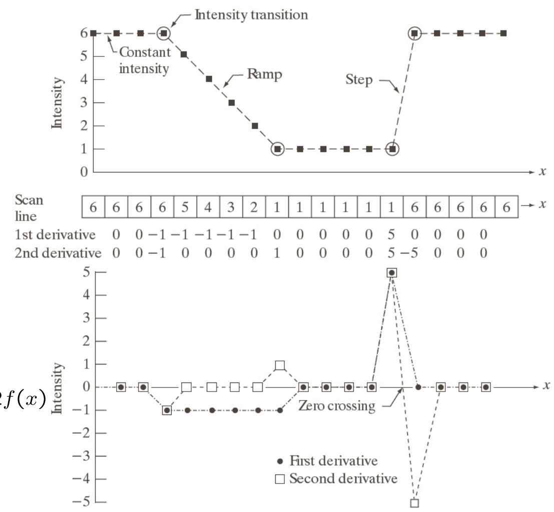
iid noise $y = x + n$ $\quad p(n=1) = p_0,\ p(n=-1) = p_0,\ p(n=0) = 1-2p_0$

more at lecture 7, "image restoration"

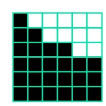
image derivative and sharpening

$$
\begin{aligned}
f'(x) &= \frac{\partial f}{\partial x} \\
&= f(x+1) - f(x)
\end{aligned}
$$

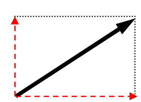
$$
\begin{aligned}
f''(x) &= \frac{\partial^2 f}{\partial x^2} \\
&= \frac{\partial f}{\partial x}(f'(x) - f'(x-1)) \\
&= f(x+1) + f(x-1) - 2f(x)
\end{aligned}
$$



edge and the first derivative



- Edge: pixel locations of abrupt luminance change
- Spatial luminance gradient vector
  - a vector consists of partial derivatives along two orthogonal directions
  - gradient gives the direction with highest rate of luminance changes



- Representing edge: edge intensity + directions
- Detection Methods
  - prepare edge examples (templates) of different intensities and directions, then find the best match
  - measure transitions along 2 orthogonal directions

edge detection operators

Image gradient:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

$$\|\nabla f\| \approx |G_x| + |G_y|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Robert's operator

−1	0
0	1

0	−1
1	0

Sobel's operator

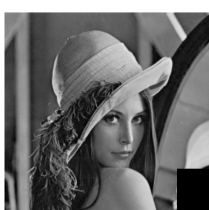
−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

a
b c
d e

FIGURE 3.41 A 3 × 3 region of an image (the *z*s are intensity values). (b)–(c) Roberts cross gradient operators. (d)–(e) Sobel operators. All the mask coefficients sum to zero, as expected of a derivative operator.

edge detection example



Roberts

Sobel

http://flickr.com/photos/reneemarie11/97326485/

-49-

second derivative in 2D

Image Laplacian: $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\Rightarrow \quad \nabla^2 f = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

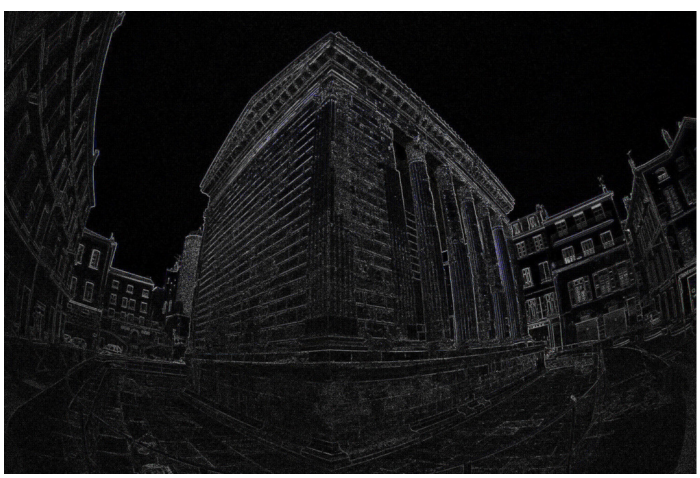
−1	−1	−1
−1	8	−1
−1	−1	−1

a b
c d

FIGURE 3.39 (a) Filter mask used to implement the digital Laplacian, as defined in Eq. (3.7-4). (b) Mask used to implement an extension of this equation that includes the diagonal neighbors. (c) and (d) Two other implementations of the Laplacian.

-50-

laplacian of roman ruins



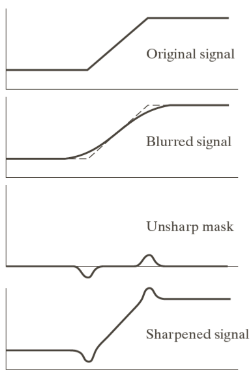
http://flickr.com/photos/starfish235/388557119/

unsharp masking

- **Unsharp masking** is an image manipulation technique for increasing the apparent sharpness of photographic images.
- The "unsharp" of the name derives from the fact that the technique uses a blurred, or "unsharp", positive to create a "mask" of the original image. The unsharped mask is then combined with the negative, creating a resulting image sharper than the original.

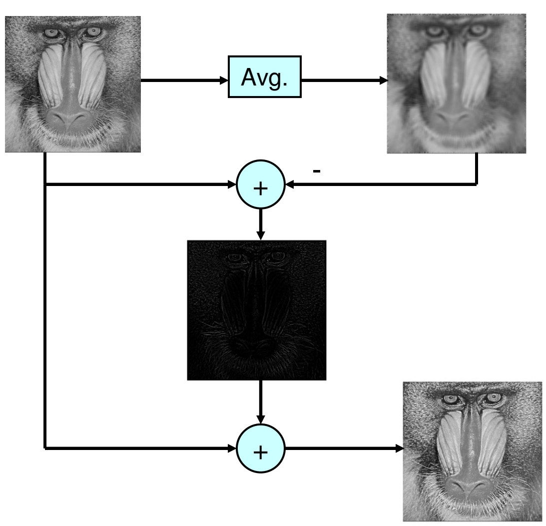
a
b
c
d

FIGURE 3.39 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).



- Steps
  - Blur the image
  - Subtract the blurred version from the original (this is called the *mask*)
  - Add the "mask" to the original

high-boost filtering



0	−1	0
−1	A + 4	−1
0	−1	0

−1	−1	−1
−1	A + 8	−1
−1	−1	−1

$$f_{hb}(x, y) = Af(x, y) - f_{lp}(x, y)$$

Unsharp mask:

high-boost with A=1

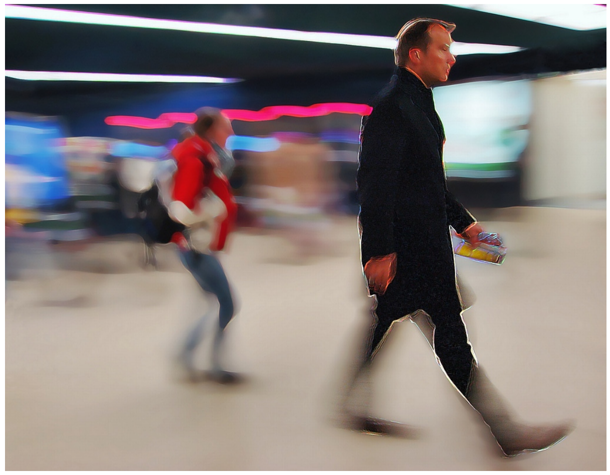
unsharp mask example

summary

- Spatial transformation and filtering are popular methods for image enhancement
- Intensity Transformation
  - Intensity transformation functions (negative, log, gamma), intensity and bit-place slicing, contrast stretching
  - Histograms: equalization, matching, local processing
- Spatial Filtering
  - smoothing filters, sharpening filters, unsharp masking, laplacian
- Combining spatial operations (sec. 3.7)

-55-

sharpen !

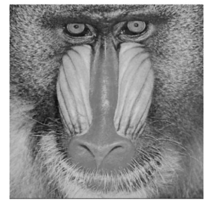


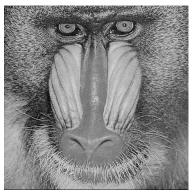
http://flickr.com/photos/t_schnitzlein/87607390/

-56-

order statistics filters

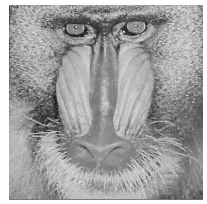
$$g(x,y) = \underset{(s,t)\in W_{(x,y)}}{median}\{f(s,t)\}$$





original

$$g(x,y) = \max_{(s,t)\in W_{(x,y)}}\{f(s,t)\}$$



$$g(x,y) = \min_{(s,t)\in W_{(x,y)}}\{f(s,t)\}$$